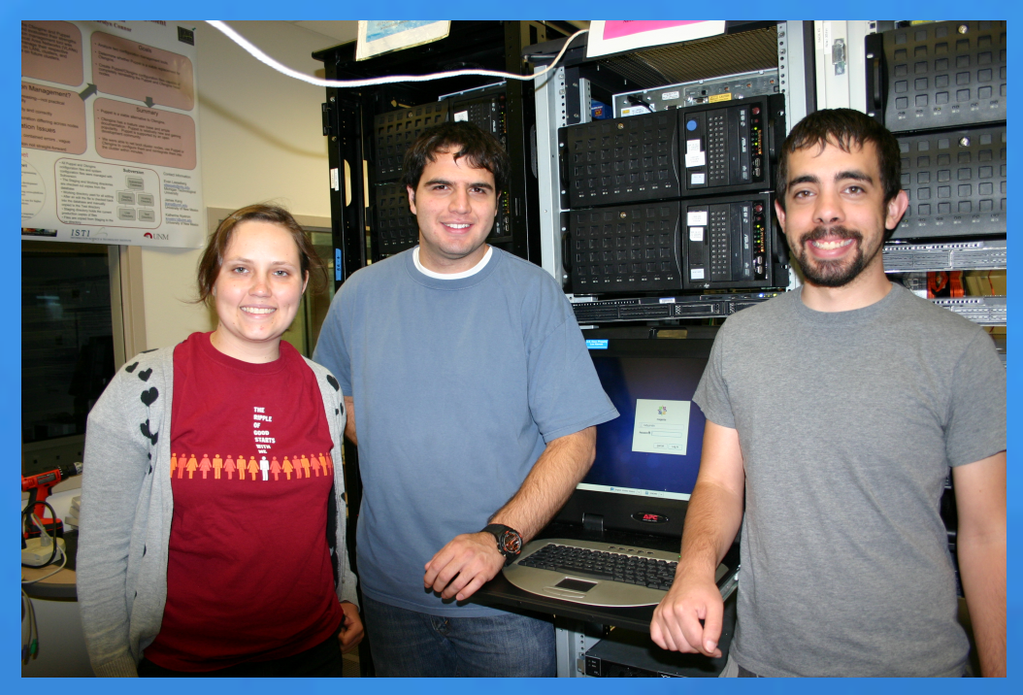
iSSH v. Auditd: Intrusion Detection in High Performance Computing

Computer System, Cluster, and Networking Summer Institute

David Karns, New Mexico State University
Katy Protin, The University of North Carolina at Chapel Hill
Justin Wolf, California State University, San Bernardino



Abstract

Our goal in this project was to provide insight into intrusions in high performance computing, focusing on tracking intruders' motions through the system. Currently, pattern-matching tools are used to detect suspicious behavior, but these tools do not provide methods to track a hacker's motions once inside the system. We tested two tools, instrumented secure shell (iSSH) and the Linux Auditing Framework (Auditd) to see if they provided insight into whether a users behavior is malicious. We wanted to explore how each tool is implemented, which is more effective, and how they affect computer performance. While doing this project, we worked to modify these tools so that they would catch more types of suspicious behavior. We tested each tool by attacking our computer cluster, then modifying them to catch each type of attack. In this project, we found that both tools have both limitations and strengths. iSSH is most useful for tracking keystrokes and reporting suspicious commands, but as a newer tool, it is not as well documented and was difficult to set up. Auditd is good for keeping daily logs of activities that can be (but are not necessarily) malicious, but does not have as many reporting capabilities as iSSH. While conducting performance testing, we noticed that the rate of file transfer using SCP increased with iSSH. We did not observe any difference in network performance with Auditd or iSSH, but we would like to perform more extensive test given future opportunity.

Process

Our computer cluster was made up of eight nodes: one head node and seven child nodes. The head node had CentOS 6.3, a Red Hat Linux operating system, installed. This node acted as the server for many services on the child nodes, such as DNS, DHCP, NTP, NFS, HTTP, and LDAP. Once iSSH and Auditd were installed, this was also where the all of the logs were sent to. Additionally, for iSSH, this was where the logs were analyzed and turned into Bro events.

Our child nodes had CentOS 5.2 installed and were configured for many of the services hosted by the head node. On some of them, we used PXELinux to netboot after creating a kickstart script; on others, we used a hard drive installation. Installing an older operating system on these nodes gave us a less secure environment, which was easier to attack. We ran a Nessus vulnerability scan on one of these nodes to give us an idea of where to start our attacks. However, the older operating system also led to some configuration problems throughout our project, such as when we were trying to install Ganglia for node monitoring.

Once our cluster was set up, we began to install and configure iSSH and Auditd. Auditd was already installed with the operating system, but it was not configured to catch many attacks. iSSH had to be downloaded and built, then configured to work with the Bro Intrusion Detection System. After installation and configuration, we modified these tools so that they would catch more types of suspicious behavior. Auditd had the capability to add rules to record various user behaviors, while iSSH had lists of suspicious commands that could be modified. We then tested each tool by attacking our computer cluster, then modifying again so that each tool would more effectively notice and respond to different types of attacks.

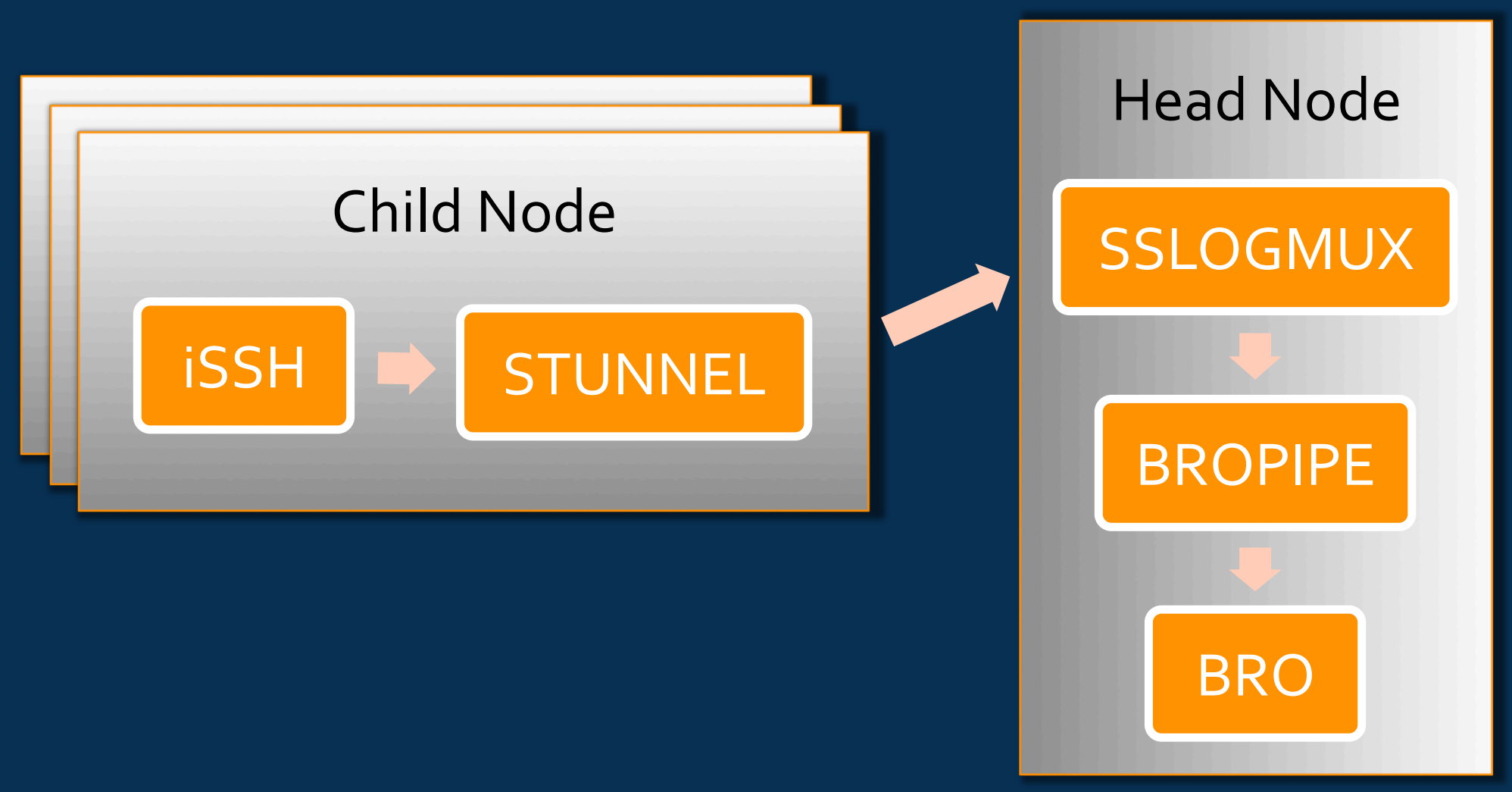
Attack Methods

Our attacks mainly focused on root privilege escalation, where an attacker with normal privileges is attempting to gain more control over the system. During these attacking drills, we assumed that the hacker has access to the system as a regular user. If they are able to gain root privileges, they will be able to modify files and perform commands that regular users cannot.

We also constructed attacks where the hacker takes advantage of improper file permissions and operating system vulnerabilities. Many files, such as those that run automatically every hour or that contain data about passwords, need to be configured so that only root users can modify them, but often file permissions are misconfigured.

iSSH

The first tool we tested is known as Instrumented Secure Shell. This is a version of SSH, which is a tool that allows users to remotely login to computer systems, that was recently developed at Lawrence Berkeley National Laboratory. Its goal is to audit user activity within a computer system to increase security. In order to accomplish this, on the client side, iSSH records all activity during an SSH session, creating a new record for each login. Then stunnel, or SSL tunnel, encrypts the data gathered and sends it to the head node. On the head node, this data is received and decrypted by the SSLLogMux, and sent to the Bropipe. Bropipe acts as a communication channel to feed this data to a running Bro instance which creates events, possibly causing alerts.



We made several modifications to iSSH. First, we wanted Bro to send us emails when something suspicious happened, such as when a suspicious command was executed locally or remotely, or an unauthorized user attempted to access the system. We added more commands to the list of suspicious commands, including useradd (creates a new user), mkdir (creates a new directory), and nc (uses netcat to open a port between machines). Finally, we experimented with the suspicious command threshold, which was set initially to five.

Auditd

The second tool we tested was Auditd, the user component of the Linux Auditing System. This service creates logs of user behavior and monitors systems calls and file accesses. It aims to improve system security by tracking users' actions within the system. During our project, we configured Auditd on each of the nodes. In a large production environment, with thousands of nodes, it would be too time consuming to monitor each node's log individually, so we sent them all to the head node.

We modified Auditd by adding rules to record different types of user behavior. After these modifications, Auditd could record events that modified user's accounts and changed network settings. It monitored users logging in and out of the system, information which is essential in tracking possibly compromised accounts. It also monitored files for permission changes and deletions. The final changes we made allowed Auditd to record new process initiations and unauthorized access attempts.

Instructor:
Dane Gardner
Mentors:
Alex Malin, HPC-DO
Graham Van Heule, HPC-3
Jim Williams, HPC-3

LA-UR-12-23587

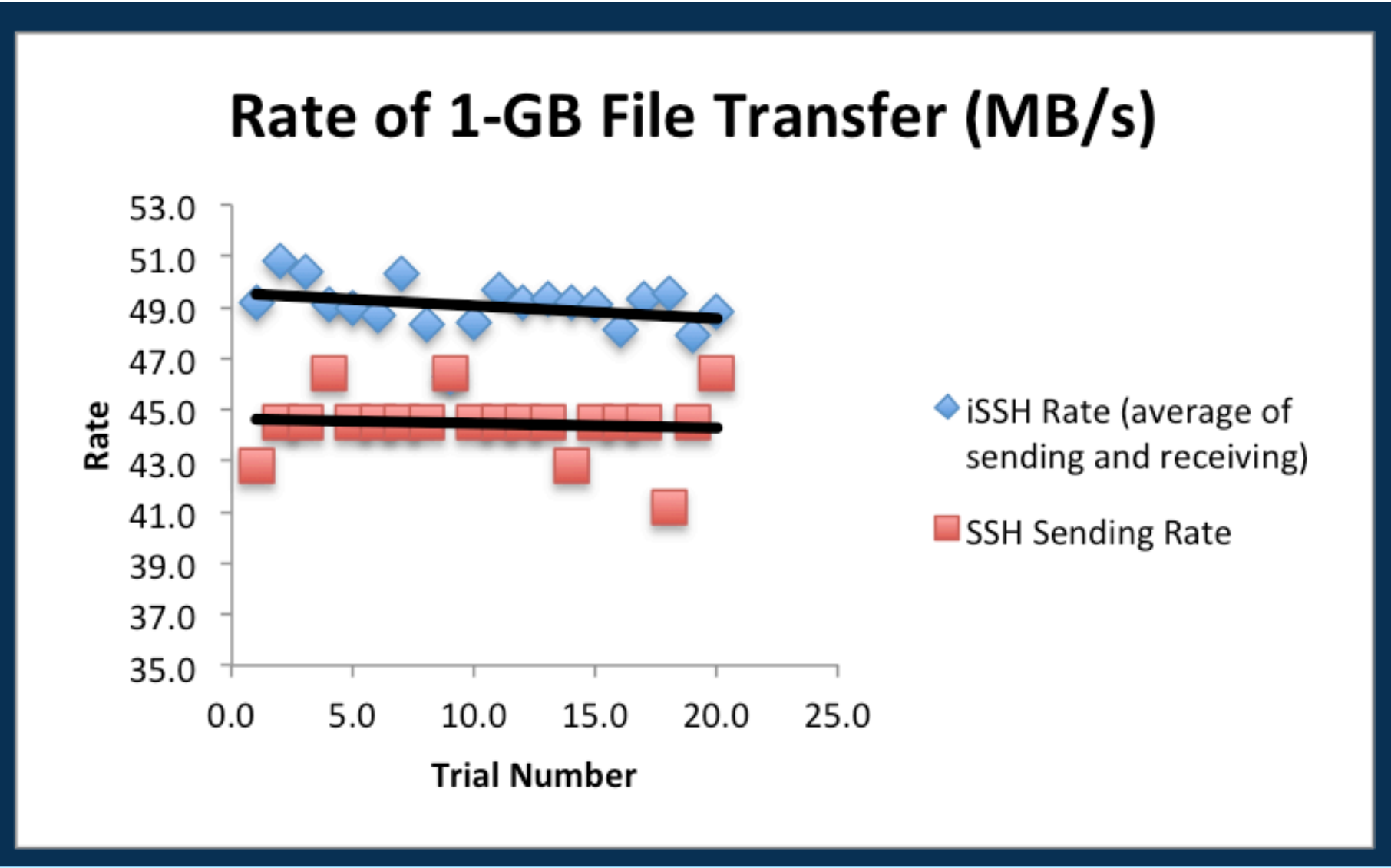
Performance Testing

TCP Throughput While Idle (10 Mb/s)			
	Control	iSSH	Auditd
Average	941.42	941.42	941.42
Std. Dev.	0.0	0.0	0.0

UDP Receive Throughput While Idle (10 Mb/s)			
	Control	iSSH	Auditd
Average	958.63	961.105	961.6
Std. Dev.	2.1	0.3	0.0

TCP Throughput While User Activity (10 Mb/s)			
	Control	iSSH	Auditd
Average	941.41	941.15	941.41
Std. Dev.	0.0	0.1	0.0

UDP Receive Throughput While User Activity (10 Mb/s)			
	Control	iSSH	Auditd
Average	961.18	959.13	960.64
Std. Dev.	0.5	0.8	0.6



Conclusion

While setting up and testing iSSH and Auditd, we observed that both of the tools have their strengths and weaknesses. Auditd is better documented than iSSH, which would help administrators during set up and troubleshooting. However, Auditd can easily overflow the computer with the volume of its logs, partly because of the high level of false alarms. In contrast, iSSH has a cleaner notification system because of its integration with Bro, but the logs are not as detailed as Auditd. Its keystroke logging capabilities allow it to easily catch suspicious commands.

From our performance testing, we found that SCP (used to copy files between machines) speed is increased when using iSSH. We also performed network tests using the Netperf benchmarking tools while Auditd or iSSH were running. We tested TCP throughput and UDP receive throughput and found that they were roughly the same regardless of which tool was running. We also found that pinging took about the same time for both tools. These results show that iSSH and Auditd don't slow down the network while running; however we must remember that the nodes were otherwise idle during this testing period, so they weren't busy responding to security threats.

Future Work

To continue our work on this project, we would like to duplicate some more of the tests done at Lawrence Berkeley National Lab with iSSH, such as testing the speed of remotely executed commands. It could be useful to repeat our network test while iSSH and Audit are experiencing more extensive user activity instead of idling. Also, we would like to experiment with more ways to expand iSSH, such as triggering different types of Bro events. With Audit, we would like to create a way to narrow down the logs it produces and enable a notification system similar to that of iSSH.

Los Alamos NATIONAL LABORATORY EST.1943
New Mexico CONSORTIUM
NSF
CALIFORNIA STATE UNIVERSITY SAN BERNARDINO
NM STATE UNIVERSITY
UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL
PRObE Parallel Reconfigurable Observational Environment
ISTI INFORMATION SCIENCE & TECHNOLOGY INSTITUTE